INFLUENCIA DE LOS TEMAS IMPARTIDOS EN EL MINOR “DESARROLLO.NET FRAMEWORK” EN MI EXPERIENCIA LABORAL

INTRODUCCIÓN

El proceso del desarrollo de software requiere de la combinación de diferentes tecnologías y plataformas con el objeto de alcanzar la funcionalidad requerida, poniendo en manifiesto la relevancia de un estudio específico de su estructura.

Dentro de esta orientación se enmarca el creciente interés por el desarrollo de software, dando lugar a la importancia de adquirir conocimientos a través de la realizacion de cursos de actualizacion, diplomados, especializaciones, entre otros, con el fin de adquirir la bases necesarias para aprender lenguajes de programacion con bases solidas, y entender la forma de diseñar una arquitectura de software a partir del uso de patrones de desarrollo y buenas practicas en el desarrollo de sistemas de información.

Este documento tiene el objeto de exponer como mi experiencia laboral, luego de realizar el Minor en Desarrollo .NET en la Universidad Tecnológica de Bolivar fue de gran influencia en mi quehacer diario como Lider y Desarrollador de Software. A traves de tres capitulos se describen las bases conceptuales aprendidas, su aplicación en mi labor diaria y los aportes que trajeron consigo, en los productos desarrollados a mi cargo.

CAPITULO 1

MINOR DE DESARROLLO EN .NET FRAMEWORK

El Minor en Desarrollo en .NET Framework, que ofrece la Universidad Tecnologica de Bolivar de acuerdo a informacion en su sitio web, encierra los aspectos representativos y novedosos de los ambientes distribuidos con aplicación práctica en plataformas de tecnologías .NET framework, con el objetivo de dar a conocer los aspectos representativos y novedosos de arquitecturas orientadas a servicios en plataformas de tecnologías .NET, y desarrollar soluciones empresariales utilizando las nuevas técnicas de diseño e implementación de aplicaciones Web con un enfoque orientado a servicio utilizando framework de .NET.

A continuación, se muestran los temas presentes dentro del plan de estudio del Minor con el fin de mostrar la base conceptual aprendida.

1.1 Ingeniería de Software

El Minor en Desarrollo en .NET Framework, se encuentra inmerso en el área de la Ingenieria de Software, razón fundamental para revisar algunos topicos generales, con el fin de entender cómo es el ciclo de vida del desarrollo de software, metodologias de desarrollo, metodologías agiles y patrones de diseño de software.

1.1.1 Patrones de Diseño

De acuerdo a (Gamma, et al., 1995) define el concepto de patrón de diseño como “soluciones simples y elegantes a problemas específicos en el diseño de software orientado a objetos”. Los patrones de diseño dan un mecanismo codificado para describir problemas y su solución de forma tal, que permiten a la comunidad poder reutilizar el conocimiento. Un patrón describe un problema, indica el contexto y permite que el usuario entienda el ambiente en el que sucede el problema, y enlista un

sistema de fuerzas que muestra cómo puede interpretarse el problema en su contexto, y en el modo en el que se aplica la solución.

Dentro de la tematica de patrones de diseño, tambien se encuentran los patrones arquitectónicos, los cuales describen problemas de diseño amplios que se resuelven con un enfoque estructural; patrones de datos, que describen problemas recurrentes orientados a datos y las posibles soluciones para modelarlos; patrones de componentes, los cuales son asociados a problemas en el desarrollo de subsistemas y componentes, de manera que se comunican entre sí y se ubiquen en una arquitectura mayor, y patrones de diseño de interfaces que describen problemas comunes de la interfaz de usuario y su solución que incluye las características específicas de los usuarios finales.

1.1.2 Métodologias de Desarrollo de Software

De acuerdo a (Carrillo, et al., 2008), "las Metodologías de Desarrollo de Software surgen ante la necesidad de utilizar una serie de procedimientos técnicas herramientas y soporte documental a la hora de desarrollar un producto de software. Dichas metodologías pretenden guiar a los desarrolladores al crear un nuevo software, pero los requisitos de un software a otro son tan variados y cambiantes, que ha dado lugar a que exista una gran variedad de metodologías para la creación del software".

1.1.3 Scrum

Scrum es un marco de trabajo para la administración y desarrollo de proyectos de software que reusa varias tecnicas de las metodologias agiles de desarrollo y algunas practicas del método JIT[1]. Con Scrum se espera conseguir resultados anticipados, flexibilidad, retorno de inversión, mitigación de riesgos, productividad y calidad, alineamiento entre cliente y un equipo de trabajo motivado.

Scrum es iterativo, cada iteración se conoce con el nombre de Sprint, se recomienda que su duración no sea mayor de un mes ni inferior a una semana.

[1] Just In Time. También conocido como el método Toyota. No debe confundirse con la compilación JIT

Roles en Scrum:

Product Owner: Representa a la voz del cliente es directamente responsable por el producto desde la perspectiva del negocio.
Scrum Master: Responsable de llevar a cabo el sprint, eliminando cualquier obstaculo que se presente ante el equipo de trabajo.
Equipo de trabajo: Equipo con las habilidades necesarias para desarrollar el producto. Se recomienda entre 3 y 8 miembros.

Reuniones:

- **Daily Scrum**
  Todos los días hay una reunión de seguimiento llamada Daily Scrum, esta se lleva de pie (está demostrado que en una reunión donde todos los miembros estén de pie todo fluye con mayor rapidez) y en ella cada miembro del equipo debe responder las siguientes preguntas ¿Qué hice ayer?, ¿Qué voy a hacer hoy?, ¿Qué inconveniente tuve al realizar mi trabajo ayer?
- **Reunión de planificación del Sprint**
  Se estima el tiempo, alcance y el detalle del trabajo que se realizará en el sprint. Debe tardar maximo ocho horas.
- **Reunión de revisión del Sprint**
  Se presenta el trabajo terminado a los interesados, tiene un limite maximo de cuatro horas.
- **Retrospectiva**
  Cada miembro del equipo responde a las preguntas. ¿Qué se hizo bien?, ¿Qué no se hizo bien?, ¿Cómo podemos mejorar lo que no se hizo bien?. El Scrum Master es responsable de dar seguimiento a los puntos a mejorar que se trataron en la reunión.

1.2 Introducción a Microsoft .NET Framework y el lenguaje de progración C#

En este módulo se explica brevemente los componentes y caracteristicas generales del Microsoft .NET Framework, su proceso de compilación y ejecución de una aplicación programada en .NET; las competencias necesarias y suficientes para escribir, compilar, depurar y ejecutar código en C# .NET.

1.2.1 Definición y Caracteristicas Generales del Micorsoft .NET Framework

.NET es un Marco de Trabajo desarrollado por Microsoft Corporation. En la documentación oficial de la MSDN[2], Microsoft (2015) lo define de la siguiente forma: "Un entorno de ejecución administrado que proporciona diversos servicios a las aplicaciones en ejecución. Consta de dos componentes principales: Common Language Runtime (CLR), que es el motor de ejecución que controla las aplicaciones en ejecución, y la biblioteca de clases de .NET Framework, que proporciona una biblioteca de código probado y reutilizable al que pueden llamar los desarrolladores desde sus propias aplicaciones."

El Framework provee los siguientes servicios a las aplicaciones en ejecución.

i. Administración de la memoria.
ii. Sistema de tipos comunes.
iii. Biblioteca de clases extensa.
iv. Interoperabilidad de lenguajes.
v. Compatibilidad de versiones.
vi. Ejecución en paralelo.
vii. Compatibilidad con múltiples versiones de Windows.

[2] Microsoft Developer Network https://msdn.microsoft.com

1.2.2 Esquema de compilación y ejecución del .NET Framework

El código fuente escrito en lenguajes de alto nivel[3] es procesado por un compilador que genera un ensamblado que contiene código CIL[4]. El entorno de ejecución del Framework es el CLR[5], este es capaz de interpretar, procesar y compilar el código CIL usando una tecnica de compilación dinamica comunmente conocida cómo JIT, el CLR se encarga de administrar eficientemente la memoria y genera código nativo que es interpretado directamente por el Sistema Operativo.

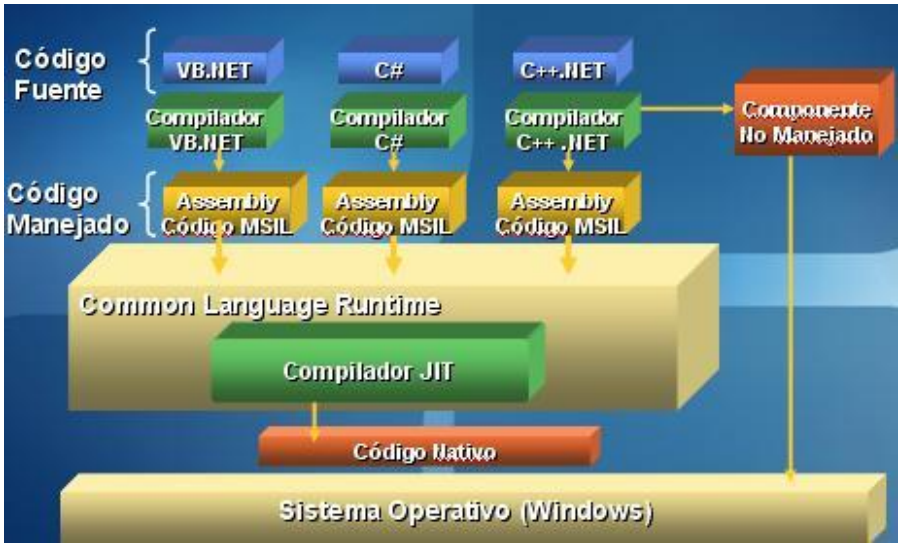

Figura 1. Esquema de compilación y ejecución del .NET Framework

1.2.3 Caracteristicas especializadas de C# .NET

C# es un lenguaje de programación diseñado por Microsoft Corporation desde el año 2000. El lenguaje C# deriva su sintaxis de C/C++ y es considerado multiparadigma (estructurado, imperativo, orientado a objetos, dirigido por eventos, funcional, genérico y reflexivo).

A continuación, se listan algunas caracteristicas que brinda C# .NET cómo lenguaje de programación, tomadas de la documentacion de Microsoft en MSDN:

---

[3] .NET Framework es potencialmente compatible con cualquier lenguaje de alto nivel que implemente el sistema de reglas del Lenguaje de Especificación Común (en inglés conocido como CLS siglas de Common Language Specification)
[4] Common Intermediate Language (Lenguaje Común Intermedio)
[5] Common Language Runtime

- **Tipos Genericos**

Los tipos genericos estan disponibles desde la versión 2.0 del Framework, encargados de implementar el concepto de parámetros de tipo, los cuales permiten diseñar clases y métodos que difieren la especificación de uno o más tipos hasta que el código de cliente declara y crea una instancia de la clase o del método.

- **Tipo Dinámico**

Se encuentra disponible desde la versión 4.0 del Framework, omitiendo la comprobación de tipos en tiempo de compilación. En tiempo de compilación, un elemento de tipo dynamic admite cualquier operación, si el código no es válido, los errores se detectan en tiempo de ejecución.

- **LINQ**

LINQ es un conjunto de características que facilitan consultas en colecciones, incorporando patrones para consultar y actualizar datos. Es extensible con cualquier tipo de almacén de datos y se encuentra disponible desde la versión 3.5 del Framework.

- **Clases Parciales**

Es posible dividir la definición de una clase, estructura o interfaz en multiples archivos de código fuente. Cada archivo de código fuente contiene una sección de la clase donde se definen atributos, propiedades y/o métodos.

1.3 Desarrollo ASP.NET con WebForms y MVC

En el módulo de WebForms y MVC se desarrollaron competencias basicas en el desarrollo de aplicaciones web utilizando tecnología .NET y tecnologias derivadas de ASP.NET.

1.3.1 Definición de ASP.NET

En la documentación oficial de la MSDN se define de la siguiente forma: “Es una plataforma web que proporciona todos los servicios necesarios para compilar aplicaciones web empresariales basadas en servidor. ASP.NET está compilado en .NET Framework, por lo que todas las características de .NET Framework están disponibles en las aplicaciones ASP.NET”.

1.3.2 Ciclo de vida de una página ASP.NET

Cuando una página generada desde una aplicación ASP.NET alojada en Internet Information Service es invocada desde un navegador pasa por las siguientes fases.

- **Solicitud de página**

La solicitud de página se produce antes de que comience el ciclo de vida de la página. Cuando un usuario solicita la página, ASP.NET determina si ésta se debe analizar y compilar (a fin de que comience el ciclo de vida de la página) o si se puede enviar una versión en caché de la página como respuesta sin ejecutar la página.

- **Inicio**

En el paso de inicio, se establecen las propiedades de la página, como Request y Response. En esta fase, la página también determina si la solicitud es una devolución de datos o una nueva solicitud, y establece la propiedad IsPostBack. Además, durante esta fase se establece la propiedad UICulture de la página.

- **Inicialización de página**

Durante la inicialización de la página, los controles incluidos en ella están disponibles y se establece la propiedad UniqueID de cada uno de ellos. Además, se aplican los temas correspondientes a la página. Si la solicitud actual es una devolución de datos, los datos de devolución aún no se han cargado y los valores de las propiedades del control no se han restaurado a los valores del estado de vista.

- **Carga**

Durante la carga, si la solicitud actual es una devolución de datos, las propiedades del control se cargan con información recuperada del estado de vista y del estado del control.

- **Validación**

Durante la validación, se llama al método Validate de todos los controles de validación, que establece la propiedad IsValid de cada uno de los controles de validación y de la página.

- **Control de eventos de devolución de datos**

Si la solicitud es una devolución de datos, se llama a los controladores de eventos.

- **Representación**

Durante la representación, el estado de vista se guarda en la página y, a continuación, ésta llama a cada uno de los controles para que aporte su salida representada al valor OutputStream de la propiedad Response de la página.

- **Descarga**

Se llama a la descarga cuando la página se ha representado completamente, se ha enviado al cliente y está lista para ser descartada. Llegado este momento, se descargan las propiedades de la página, como Response y Request, y se llevan a cabo las operaciones de limpieza correspondientes.

1.3.3 ASP.NET MVC Framework

ASP.NET MVC es un Framework para el desarrollo de aplicaciones web que aplica el patron de diseño Modelo Vista Controlador usando la plataforma ASP.NET.

1.3.3.1 Patron de diseño MVC

Actualmente, es uno de los patrones de diseño mas importantes y usados en la web, derivado del patron Thing-Model-View-Editor publicado en 1979. MVC separa la interfaz de usuario en tres capas.

Modelo: Conjunto de clases que contienen el acceso a datos y las reglas de negocio

Vista: Define cómo la aplicación debe verse ante un usuario final

Controlador: Orquesta el flujo general de la aplicación

1.3.3.2 Implementación de ASP.NET MVC

En 2009, Microsoft liberó la primera versión estable de ASP.NET MVC (versión 1.0) donde por primera vez existia un Framework oficial de Microsoft para trabajar MVC; desde esta implementación se promovian los controles HTML y no los controles ASP, sin embargo aun existian las clases asociadas al codebehind de los archivos ASPX. En la versión 2, se implementaron plantillas personalizables[6], soporte para llamado asincronos al controlador y se eliminaron por completo los archivos de codebehind. En la versión 3, se mejoró la integración con Jquery (especialmente con Jquery AJAX), se agregaron filtros de validación rapida, se realizó una integración con el ORM Entity Framework y se creo un motor de vistas llamado Razor, que resulta mas legible que ASP clasico.

1.4 Acceso a bases de datos SQL Server desde .NET Framework

En este modulo realizaron una serie practicas relacionadas con el acceso a datos desde las aplicaciones y modificaciones de los objetos de base de datos (tablas, disparadores y procedimientos almacenados)

1.4.1 SQL Server

Es un motor de base de datos producido por Microsoft Corporation basado en el modelo relacional, que soporta T-SQL y ANSI QSL, transacciones y procedimientos almacenados y es facil de acceder desde aplicaciones escritas en .NET Framework. Su primera versión fue liberada en 1989.

---

[6] Este cambio se hizo a nivel de componente para el Visual Studio

A continuación se describen algunos objetos de base de datos soportados por SQL Server

i. Tablas: Es el tipo de objeto mas usado de SQL en estos se almacenan los datos, generalmente recogidos por una aplicación.
ii. Procedimientos Almacenados: es un objeto que almacena una sección de código SQL físicamente en la base de datos. La ventaja de un procedimiento almacenado es que al ser ejecutado, en respuesta a una petición de usuario, es ejecutado directamente en el motor de bases de datos, el cual usualmente corre en un servidor separado.
iii. Desencadenadores: son objetos que se ejecutan en respuesta a una variedad de eventos en la base de datos.

1.5 Herramientas de desarrollo

Este modulo abarco algunas herramientas de apoyo al desarrollo de software, las cuales son detalladas a continuacion de acuerdo a la documentacion ofrecida por Microsoft en su sitio oficial.

1.5.1 Visual Studio

Es la IDE oficial para el desarrollo de .NET, corresponde a un conjunto de herramientas y otras tecnologías de desarrollo de software basado en componentes para crear aplicaciones eficaces y de alto rendimiento.

1.5.2 Team Foundation Server

Es la plataforma de colaboración y administración del ciclo de vida de las aplicaciones de Microsoft. Se integra facilmente con Visual Studio (a partir de la versión 2008).

1.5.3 Sql Management Studio

Es una aplicación que facilita la administración de SQL Server, disponible desde la versión 2005 de SQL Server.

1.5.4 SQL Server Profiler

Es una herramienta de seguimiento de SQL Server, posee una interfaz enriquecida para crear y administrar seguimientos, analizar y reproducir resultados de seguimiento. Los eventos se guardan en un archivo de seguimiento que posteriormente se puede analizar o usar para reproducir una serie de pasos específicos cuando se intenta diagnosticar un problema.

CAPITULO 2

EXPERIENCIA LABORAL EN DESARROLLO DE SOFTWARE

2.1 Universidad de San Buenaventura

Entre el noviembre de 2010 y diciembre de 2011, trabajé en la Universidad San Buenaventura seccional Cartagena cómo Desarrollador Senior en el Centro de Educación Virtual. Este espacio fue creado para abrir nuevas carreras a través de medios virtuales.

Los primeros programas virtuales creados fueron: “Formación Creativa de la Producción Multimedia” y “Tecnología en la Gestion de la Producción Multimedia”, la creación de estos programas fue en convenio con el Ministerio de Educación Nacional.

Las principal responsabilidad del cargo, era la creación de un portal académico que permitierá la gestion de notas, inscripción de cursos y proceso de matricula. El equipo directamente responsable del desarrollo estaba conformado por el director de tecnologias del Centro de Educación Virtual, un desarrollador Senior, un desarrollador Junior, tres practicantes y un diseñador Grafico. El sistema debia usar la base de datos del anterior sistema de notas, la cual se encontraba alojada en un motor de Oracle.

Este producto se desarrolló creando una API de Servicios REST desarrollada con Windows Communication Foundation. Esta API fue escrita en C# .NET, el acceso y manipulación de datos se realizó con Nhibernate y Sprint .NET. Se utilizó una arquitectura N-Capas y se respetaron todos los principios SOLID (Single responsibility, Open-closed, Liskov substitution, Interface segregation and Dependency inversion).

El cliente del API fue desarrollado en PHP sobre el CMS Joomla!, utilizando el motor de vistas Smarty y los beneficios de Jquery.

Después de algunos meses de haber iniciado el proyecto, se decidió comenzar a utilizar todas las practicas de Scrum y a realizar estimaciones utilizando Poker Planning; inmediatamente subió el porcentaje de productividad del equipo en relacion

con los primeros meses de trabajo. Al finalizar el alcance inicial del desarrollo se denomino al sistema de administración académica con el nombre AcaWeb-Usb.

Actualmente, AcaWeb-Usb se encuentra operando en la Universidad de San Buenaventura; y ambos programas hacen parte de la oferta de dicha universidad.

2.2 Zeus Tecnología SA

Desde julio de 2012, laboro en Zeus Tecnología SA, especificamente en el proyecto de migración del software Zeus Hoteles a la Web. Este sistema se desarrolla en C# .NET Framework 4.0; esta compuesto de multiples extensiones instaladas al sistema de administración de contenido DotNetNuke. La administración y seguimiento del proyecto se lleva a cabo usando una versión ligera del marco de trabajo Scrum.

A lo largo del tiempo laborado en Zeus Tecnología, cuento con la siguiente experiencia:

2.2.1 Ingeniero de Desarrollo Nivel 2

Con este cargo di inicio a mi actividad laboral dentro de la empresa, asumiendo la responsabilidad del cargo de acuerdo a las funcionalidades asignadas por el Scrum Master. El cargo requeria conocimientos en Scrum, HTML, CSS, Javascript, Jquery, C#, ASP .NET, SQL y eventualmente arquitectura orientada a Servicios. El desarrollo incluia análisis, codificación, documentación y correcciones ante posibles errores reportados en las pruebas funcionales del sistema.

2.2.2 Lider de Producto Zeus Hoteles Versión Web

Desde marzo de 2014, a la fecha ocupo el cargo de lider de producto, teniendo cómo responsabilidad, la administración y planeación de los Sprints, accesorar a los desarrolladores en cualquier inconveniente tecnico presentado, inspeccionar el código escrito por los desarrolladores para asegurar el cumplimiento de estandares y buenas practicas, además de cumplir funciones de Scrum Master, responder ante la gerencia general y la gerencia de desarrollo en el cumplimiento de tiempos de entrega de producto y calidad del software desarrollado.

2.3 Desarrollo de Proyectos Independientes

2.3.1 Portal y sistema de Gestión Tablero de Buques de Agencia Altamar

Sistema de administración maritima desarrollado para la Agencia Maritima Altamar, permitia administrar la agenda de llegada de Buques agenciados por la empresa. Fue codificado en C# .NET Framework 3.5, Coolite, ExtJs[7] y SQL Server 2005. La creación de este sofware fue contratado por la empresa Ziack Engine Ltda. Actualmente se encuentra en funcionamiento.

2.3.2 Sistema de Información para Importaciones, Exportaciones y Manejo de Bodegas e Inventarios para Isicomex Ltda.

Sistema para gestion de importaciones, exportaciones e inventarios desarrollada para la CIA Isicomex SA. Fue codificado en C# .NET Framework 3.5, servicios AMF e interfaz grafica en Flex. La creación de este sofware fue contratado por la empresa Ziack Engine Ltda. Actualmente no se encuentra en funcionamiento.

[7] Actualmente conocido como sencha

CAPITULO 3

APLICACIÓN DE TEMAS RELACIONADOS CON EL MINOR DE DESARROLLO .NET

Zeus Tecnología es una empresa de desarrollo de software fundada en Cartagena en 1999, actualmente ofrece varias soluciones empresariales, tales como Zeus Agencias, Zeus Contabilidad, Zeus Clubes, Zeus Inventario, Zeus Nomina, Zeus Restaurantes Zeus Salud y Zeus Hoteles. Este ultimo tal y como su nombre lo indica está dedicado a la administración hotelera; posee cientos de clientes en latinoamerica, siendo uno de los sistemas mas vendidos de la suite Zeus. El sistema se ejecuta en windows como una aplicación stand alone, ha sido desarrollada en Microsoft Visual FoxPro 9.0 con un motor de base de datos Microsoft SQL Server[8].

Buscando una actualización del sistema se realizó un proyecto para migrarlo a la web, este recibió apoyo de Colciencias y ESI Center. El proyecto inició a finales de 2011, pero hasta marzo de 2012 inició la codificacion del nuevo sistema.

La meta del proyecto era que el nuevo sistema tuvierá toda la funcionalidad de la versión stand alone y ademas una interfaz moderna, amigable e intuitiva para el usuario final. Para esto se contrató un grupo de trabajo de aproximadamente 39 personas, entre los cuales habian desarrolladores de base de datos (nivel Junior y nivel Senior), desarrolladores web (nivel Junior y nivel Senior), asesores de desarrollo, diseñadores graficos, lideres tecnicos y un director del proyecto.

Después de realizar un analisis completo de requerimientos funcionales y no funcionales del sistema, se inició una fase de investigación para así elegir las tecnologias adecuadas para el desarrollo del sistema Zeus Hoteles Web. Resultado de la investigacion realizada por el director de proyecto, en compañía de los lideres tecnicos y asesores, se tomó la decisión de desarrollar la nueva versión del sistema con la estructura y versión original de la base de datos, ofreciendo asi compatibilidad entre las versiones del sistema. La interfaz seria desarrollada en ASP.NET usando C#

[8] Versión 2008 R2 o superior

cómo lenguaje de servidor, las opciones tipo Maestros[9] serian desarrolladas con un generador de codigo parcialmente flexible conocido cómo Code On Time. Finalmente, se eligió DotNetNuke como el sistema de manejo de contenido de la software. En cuanto a la metodologia de desarrollo se eligió una variación de Scrum basada en Scrum de Scrums.

Todas las tecnologias utilizadas en el desarrollo del proyecto son tecnologias basadas en .NET, las cuales directa o indirectamente fueron tratadas con un nivel de profundización basico o intermedio en los modulos del Minor de Desarrollo .NET.

3.1 DotNetNuke

DotNetNuke (popularmente abreviado cómo DNN) es un Framework de código abierto para la administración de portales y contenido, basado en tecnología .NET de Microsoft[10]. DNN ofrece un marco robusto, extensible y completamente funcional para el desarrollo de una amplia gama de aplicaciones web.

DNN es líder mundial en Framework para la administración de portales y contenido, adoptada por miles de organizaciones en todo el mundo. Ofrece una interfaz para administración de multiples aplicaciones dentro del mismo sitio web.

Caracteristicas generales de DotNetNuke:

i. Multidioma (administración y contenido) sin tener que crear portales adiciones.
ii. Tratamiento personalizado del error 404.
iii. Configuraciones varias para posicionamiento de páginas en buscadores (opcional).
iv. Previsualización en la configuración de la página del enlace y la descripción de la página en Google.
v. URLs amigables.

[9] Opciones cuyo único propósito es listar, agregar, modificar y eliminar información de un tipo de objeto del modelo de dominio
[10] Fue escrito originalmente en Visual Basic .NET y posteriormente migrado a C# .NET

vi. Tratamiento avanzado de CSS, permite modificación de la css desde el portal (sólo para usuarios superadministradores).
vii. Permite modificación del archivo de configuración desde el sitio (sólo para usuarios superadministradores).
viii. Multiportal.
ix. Administraciones de usuarios. Roles y permisos para cada portal.
x. Flexibilidad en apariencia de las paginas (permite multiples skins).
xi. Gestion de tareas programas.
xii. Arquitectura modular (micronucleo).

3.1.1 Arquitectura Micronucleo

De acuerdo (Almeira, et al., 2007) define el patrón arquitectónico Micronucleo[11], como aquel que se "aplica a los sistemas de software que pueden adaptarse a cambios en los requerimientos, donde se separa una función mínima central de la funcionalidad extendida y partes específicas del cliente, y sirve como un socket para conectar extensiones y coordinar su colaboración."

La arquitectura Microkernel está constituida por una serie de componentes: Microkernel, servidor interno, servidor externo, clientes y adaptadores.

El Microkernel es el componente principal de la arquitectura, encargado de implementar servicios centrales: medios de comunicación, mantenimiento, control y coordinación de recursos como procesos y archivos, además de encapsular diferentes dependencias específicas del sistema.

A su vez, los Servidores Internos se conocen con el nombre de subsistemas y actúan como componentes asociados que ofrecen funcionalidades adicionales que proporciona el microkernel. Los servidores internos encapsulan algunas dependencias del hardware o software subyacente como por ejemplo: los controladores de dispositivos que soporten tarjetas gráficas específicas.

[11] También llamado microkernel

Por su parte, los servidores externos tienen el nombre de personalidades, los cuales utilizan el microkernel para implementar su propia vista de dominio de la aplicación subyacente, exportando interfaces de la misma manera que lo realiza el kernel, corriendo en un proceso separado.

Los clientes son aplicaciones accedidas a través de la interfaz de programación que proporciona el servidor externo y los adaptadores representan las interfaces entre los clientes y los servidores externos, encargados de permitir el acceso de los servicios a los clientes de forma portable. Además, son los encargados de proteger a los clientes de los detalles de implementación específicos del microkernel.

En el caso de DotNetNuke este funciona como Microkernel, y cada una de las extensiones que se instalan en él (Skin, Controles, etc) actuan cómo subsistemas.

3.1.2 Manejo de extensiones con DNN

La forma de extender las funcionalidades de CMS de DNN es a través de extensiones, con el fin de modificar la apariencia y adicionar nuevos módulos que podran ser utilizados en las páginas de los portales.

Hay varios tipos de extensiones disponibles para extender DNN:

Authentication System: Permite crear un sistema diferente para la autenticación de usuario, por ejemplo, la autenticación mediante un cuenta de Gmail. En el proyeco se creo una extensión de este tipo para mantener la compatibilidad con la versión stand alone de la aplicación.

Container: Es un elemento de diseño que contiene la disposición general de la página. En el proyeco se creo una extensión de este tipo para la maquetación de las páginas de la aplicación.

Extension Verification Service: Permite a los usuarios probar la compatibilidad de las extensiones con versiones de DNN.

Library: Permite la instalación de un compoente que puede ser compartido por varios, por ejemplo una DLL o unas imagenes. Se crearon varias extensiones de este tipo para subir recursos (imágenes, estilos adicionales y Dlls).

Modules: Permite adicionar elementos a las páginas. Es el tipo de extensión que mas se utiliza en el proyecto, ya que permite que todas las opciones de las aplicaciones usen este tipo de extensión.

Providers: Es una pieza fundamental que proporciona información / servicios al resto de la instalación de DotNetNuke. Por ejemplo MembershipProvider, FriendlyUrlProvider, DataProvider.

Skin: Es el componente del diseño visual, permite personalizar la apariencia de una o más páginas. Se han creado varias extensiones de este tipo dentro del proyecto.

Skin Object: Es un control de usuario que puede ser utilizado por cualquier skin.

3.1.3 Manejo de permisos con DNN

DotNetNuke posee reglas y permisos para cada portal, los permisos pueden ser dados o negados a usuarios especificos a un roles[12]. Los objetos sobre los que se pueden administrar roles son páginas y módulos. Existen tres niveles de permiso los cuales son:

i. Editar: Permite realizar cualquier cambio en la configuración del objeto, incluyendo la eliminación del objeto.
ii. Ver: Tiene acceso a ver el objeto.
iii. Acceso denegado: se expresa explicitamente que el Usuario y/o Rol no se posee acceso a un objeto determinado.

De forma general se pueden agrupar los usuarios en estas categorias:

i. Los usuarios “host” (superusuarios): tienen acceso nivel Editar a todas las páginas y módulos del sitio, no es posible denegarles acceso.
ii. Los usuarios con rol “Administrator”: tienen permiso nivel Editar a todas las páginas y módulos del portal donde fueron creados, no tienen acceso a las paginas de superusuarios.
iii. Cualquier otro usuario poseen al menos el rol Registered User.

[12] En este caso se dan o niegan los permisos a todos los usuarios que poseen el rol determinado.

Todas las extensiones desarrolladas en Zeus para DNN han sido escritas total o parcialmente en C# .NET para ello fueron claves los temas tratados en el módulo "Introducción a Microsoft .NET Framework y el lenguaje de progración C#" (veáse sección 1.2), asi como los temas del módulo "Desarrollo ASP.NET con WebForms y MVC" (veáse sección 1.3).

3.2 Code On Time

Code On Time es un generador de código que lee las tablas, relaciones y procedimientos almacenados de una base de datos; a partir de ellos genera una capa de lógica y una vista, esta vista permitirá funcionalidades de creación, modificación, borrado, listados, busqueda y ademas exportar datos a multiples formatos. Ofreciendo las operaciones basicas requeridas para administrar el objetos del dominio.

Las ventajas mas importante de Code On Time es la facilidad para realizar aplicaciones basicas en pocos minutos, gracias a esto se redujo el tiempo de desarrollo de la mayoría de maestros del sistema Zeus Hoteles, alrededor de 110 maestros fueron desarrollados con esta herramienta. Otra notoria ventaja es permitir exportar las aplicaciones a un sitio web (tradicional o movil), sharepoint, publicar en azure o generar un módulo de DotNetNuke, este ultimo formato de exportación fue el usado en la generación de todos los maestros.

Code On Time está basado en .NET y por tanto la unica forma de extenderlo y/o modificarlo es a través de código escrito en C# o Visual Basic. Para trabajar con el generador de código fueron claves los temas tratados en el módulo "Introducción a Microsoft .NET Framework y el lenguaje de progración C#" (veáse sección 1.2). Dadas las limitaciones de la herramienta algunas validaciones adicionales fueron desarrolladas en procedimientos almacenados de SQL, para ello fue util el tema "SQL Server" trabajado en el modulo "Acceso a bases de datos SQL Server desde .NET Framework" (veáse sección 1.4.1).

3.3 Metodología de Desarrollo

Para el desarrollo y seguimiento del proyecto de creación de la versión Zeus Hoteles Web, se adaptó un Framework derivado de Scrum, el cual es aplicable a equipos de trabajo de mas de 7 personas[13] llamado Scrum de Scrums. En Scrum de Scrums se dividen los equipos de trabajo en equipos mas pequeños, cada sub-equipo realiza la reunión de seguimiento diaria, un "embajador" de cada equipo reporta los avances generales de su equipo en otra reunión de seguimiento diaria donde asisten los embajadores y el Scrum Master.

La adaptación de esta metodología al desarrollo del proyecto eliminó los "embajadores" haciendo que el Scrum Master estuviese presente en las reuniones de seguimiento de cada sub-equipo. Cada equipo de trabajo estaria compuesto por un grupo de 3 a 5 desarrolladores y de 1 a 2 tester (sólo se realizan pruebas de funcionalidad); cada 2 o 3 equipos compartirian un diseñador grafico.

En el lanzamiento del sprint[14] el Scrum Master asigna algunos PBI[15] a desarrolladores de acuerdo a ciertas habilidades muy especificas, las otras tareas son publicadas en un lugar donde sean accesibles por todos los desarrolladores, cada tarea debe describir de forma suficientemente clara lo que se debe hacer, además debe tener un nivel de prioridad, para indicar que tareas se deben desarrollar primero a lo largo del sprint. El desarrollador acompañado del tester debe revisar cada PBI para determinar su alcance; el diseñador grafico debe listar todas las opciones que va a diseñar en el sprint con la fecha tentativa de inicio por parte del desarrollador, ya que el diseño debe estar listo antes que el desarrollador inicie cada una de las tareas.

El ciclo de vida de un PBI inicia al momento de su creación en el lanzamiento del sprint. Cuando el desarrollador inicia la codificación el PBI pasa a "Progreso". Al terminar de codificar le pasa un listado de los artefactos modificados al lider técnico, para que este inspeccione el código, en este paso el PBI está en "Inspección". Cuando el inspector (rol que asume el lider técnico) revisa que el código escrito por el

[13] Una de los requerimientos necesarios para implementar Scrum clásico es que el equipo de desarrollo no sea mayor a 7 desarrolladores
[14] Se toma un día completo para el lanzamiento de un sprint de 2 semanas.
[15] PBI: Product Backlog Item.

desarrollador cumpla con los estandares de codificación, este cambia de estado el PBI colocandolo en "Integración"[16], el integrador revisa los artefactos modificados por el desarrollador y los integra en un ambiente de pruebas. El tester realiza todas las pruebas necesarias sobre la opción(es) modificada(s), en caso que alguna prueba no sea satisfactoria el tester cambia de estado el PBI a "fallido" genera un bug documentando bajo que procesos el sistema falla. El bug es corregido por el desarrollador el cual pasa el PBI nuevamente a inspección realizando nuevamente el proceso anteriormente documentado. En caso que el desarrollo pase todas las pruebas exitosamente el tester cambiará el estado del mismo a "Satisfacción del Cliente".

La metodología SCRUM fue facilmente asimiliada gracias al conocimiento del marco de trabajo Scrum, el cual fue tratado en el módulo "Métodologias de Desarrollo de Software" (veáse sección 1.1.2.)

3.4 Otras Herramientas Utilizadas

3.4.1 Html, CSS, Javascript y Jquery

Por desarrollar una aplicación web, el uso de estas teconologias fueron de uso constante de forma directa o indirecta en el desarrollo de todas las vistas de la aplicación. Estos temas no fueron abordados en el minor, puesto que eran requisitos para cursarlo.

3.4.2 SQL Server

El motor de base de datos de la aplicación de Zeus Hoteles y DNN esta en SQL Server, a menudo se realizaron modificaciones sobre los objetos de base de dato existentes desde la versión Stand Alone de Zeus Hoteles, adicionalmente se agregaron otros

[16] En caso que no cumpla cambia el PBI de estado colocándolo nuevamente en desarrollo, indica las correcciones necesarias para que el desarrollador aplique los estándares correctos de codificación y buenas prácticas.

objetos, en su mayoria procedimientos almacenados, puesto que la mayoria de la lógica de negocio de la aplicación se encuentra en la base de datos. Fue trabajado en el minor en módulo “Acceso a bases de datos SQL Server desde .NET Framework” (veasé sección 1.4.1)

3.4.3 Sql Management Studio

Se usa como IDE de trabajo para las modificaciones sobre los objetos de base de datos. Fue trabajado en el minor en módulo “Herramientas de desarrollo” (veasé sección 1.5.3)

3.4.4 SQL Server Profiler

Se usa para hacer seguimiento a los llamados realizados a la base de datos desde la aplicación Fue trabajado en el minor en módulo “Herramientas de desarrollo” (veasé sección 1.5.4).

3.4.5 Visual Studio

Fue la IDE de desarrollo de todo el Front de la Aplicación. Fue trabajado en el minor en módulo “Herramientas de desarrollo” (veasé sección 1.5.1).

3.4.6 Team Foundation Server

Todo el versionamiento del código y el seguimiento de los sprints, tareas, bugs y casos de prueba se han desarrollado con TFS. Fue trabajado en el minor en módulo “Herramientas de desarrollo” (veasé sección 1.5.2)

CONCLUSIONES

Desde el año 2009, he participado activamente en proyectos que involucran directa o indirectamente el desarrollo de software; lo cual ha sido una oportunidad de conocer y profundizar en diversas metodologias de desarrollo, patrones de diseño, lenguajes de programación y diversos frameworks que me han enseñado a seguir las buenas practicas en mi profesión.

Hoy en día aprender un lenguaje de programación es una dificultad mínima, el gran problema es conocer el framework, el cual puede llevar años en aprenderse. Por supuesto, un desarrollador puede modelar y construir una aplicación sin seguir ningún framework conocido; considerándolo no necesario, sin embargo, a medida que la aplicación crece, un programador competente procurará seguir unas determinadas pautas que le faciliten su trabajo de desarrollo y mantenimiento.

Trabajar con el framework .NET ha sido una de las razones para considerarme exitoso en mi profesión, ya que este conjunto de tecnologias me abrio las puertas al terreno laboral y a obtener la experiencia de siete años desarrollando con esta plataforma en marcos de trabajo .NET como: Framework (versiones 3.5, 4.0 y 4.5), ASP.NET MVC (versiones 1.0, 2.0, 3.0 y 4.0), Entity Framework, Sprint .NET y el Lenguaje de Programación C#, y en cuanto a Patrones de diseño utilizar el Modelo-Vista-Controlador, Modelo-Vista-Vista-Modelo, Abstract Factory y Singleton.

Actualmente, un porcentaje alto de las grandes empresas de desarrollo de software en Colombia tienen a Microsoft como aliado en sus tecnologias, punto importante para destacarme y crecer como profesional en un futuro cercano.

Todas las razones anteriormente citadas, llevan a la conclusión que los temas tratados en el Minor de desarrollo .NET han tenido gran influencia a lo largo de mi vida laboral.

BIBLIOGRAFÍA

Almeira, A; Perez, V (2007). Arquitectura de Software: Estilos y Patrones. Facultad de Ingeniería. Universidad Nacional De La Patagonia San Juan Bosco. Argentina

Carrillo, I., Pérez, R., Rodríguez, A. (1995). Metodologia de desarrollo del software. [en linea] http://plataforma.edu.pe/pluginfile.php/246542/mod_resource/content/1/Metodologias%20de%20desarrollo%28RUP-METODOLOS%20AGILES%29.pdf

Gamma, E., Helm, R., Johnson, R., & Vlisides, J. (1995). Design Patterns Elements of Reusable Object Oriented Software. 1-358. Addison Wesley Longman Inc. USA.

Microsoft (2015). Introducción a .NET Framework [En linea]. Disponible en: https://msdn.microsoft.com/es-es/library/hh425099(v=vs.110).aspx. Fecha de consulta 15/04/2015

Microsoft (2015). Generics (C# Programming Guide) [En línea]. Disponible en: https://msdn.microsoft.com/es-es/library/0zk36dx2.aspx Fecha de consulta 15/04/2015. Fecha de consulta 15/04/2015

Microsoft (2015). VISUAL STUDIO .NET [En línea]. Disponible en: https://msdn.microsoft.com/es-es/vstudio/aa718325.aspx. Fecha de consulta 15/04/2015

Microsoft (2015). SQL Server Profiler. [En línea]. Disponible en: https://msdn.microsoft.com/es-co/library/ms181091.aspx Fecha de consulta 15/04/2015

CodeOnTime (2015). GETTING STARTED WITH CODE ON TIME. [En linea]. Disponible en: https://codeontime.com/documents/Getting-Started-With-Code-On-Time.pdf Fecha de consulta 15/04/2015

Agile Alliance (2015). SCRUM of SCRUMS [En linea]. Disponible en: http://guide.agilealliance.org/guide/scrumofscrums.html. Fecha de consulta 15/04/2015